DOCKET NO.
XACTP009

U.S. PATENT APPLICATION

FOR

SYSTEM, METHOD AND COMPUTER PROGRAM PRODUCT FOR NETWORK RECORD SYNTHESIS

Inventor(s): Tal Givoly
Limor Schweitzer

ASSIGNEE: XACCT TECHNOLOGIES, INC.

KEVIN J. ZILKA
PATENT AGENT
P.O. BOX 721120
SAN JOSE, CA 95172

SYSTEM, METHOD AND COMPUTER PROGRAM PRODUCT FOR NETWORK RECORD SYNTHESIS

RELATED APPLICATION(S)

The present application is a continuation-in-part of an application filed 11/18/99 under serial number 09/442,876, which is incorporated herein by reference in its entirety. The present application also claims priority from a provisional application filed 10/23/00 under serial number 60/242,733, which is incorporated herein by reference in its entirety.

FIELD OF THE INVENTION

The present invention relates to data records, and more particularly to records reflecting various services afforded utilizing a network.

BACKGROUND OF THE INVENTION

Network accounting involves the collection of various types of records while sending and receiving information over a network. Examples of such records may include, but are not limited to a session's source, destination, user name, duration, time, date, type of server, volume of data transferred, etc. Armed with such accounting records, various services may be provided that require network usage metering of some sort.

Prior art Figure **1** is a diagram illustrating the various services **100** offered over a network such as the Internet. As shown, the various services include a hypertext transfer protocol (HTTP) session **102**, an electronic mail session **104**, and a voice over Internet Protocol (IP) session **106**. The HTTP session **102** may include domain browsing, accessing an HTML page/frame, etc. The electronic mail session **104** may involve an SMTP transmitting server address and a POP3 receiving server address.

As shown, various levels of detailed information may be collected regarding numerous services afforded utilizing the Internet. Such information may be gathered with varied granularity depending on the accounting purpose at hand. For example, billing is currently handled based on a dial-up session which may include all of the above sessions. There may be situations where it is desired that billing be carried out as a function of domain browsing, an email session, etc. Such types of accounting are becoming increasingly important since permanent connections, i.e. DSL, Cable, GPRS, LAN, etc., are more and more prevalent, thus rendering billing based on a dial-up session obsolete.

With such opportunities to use such diversified records of Internet usage, a problem arises in collecting the same in an organized manner. Prior Art Figure **2** illustrates prior art methods of organizing accounting information. As shown, a tailored single service data block **210** may be customized to account for specific services. For example, account, start time, duration, service, direction, quality-of-service (QoS), byte amount, number of e-mail, number of attachments identifiers or the like may be specifically accounted for in pre-allocated portions of the data block **210**.

In yet another example, a generic single service data block **212** may be used to account for common information, i.e. an account identifier, start time, duration, service identifier, etc. Further, certain portions of the data block **212** may be allocated for attributes that may vary from service to service.

While these methods of accounting for network usage are somewhat effective, they fail to allow versatility to be introduced into the tracking process. In particular, as the amount of available tracking data increases, so does the complexity in handling such accounting information. An example of such complexity is exhibited when trying to organize services provided to a single customer.

There is therefore a need for a technique of rolling up service accounting information in a way which permits improved versatility and performance.

Brief Description of the Drawings

Prior art Figure **1** is a diagram illustrating the various services offered over a network such as the Internet;

Prior Art Figure **2** illustrates prior art methods of organizing accounting information;

Figure **3** illustrates a method for generating a single record reflecting multiple services for accounting purposes;

Figure **4** illustrates an exemplary network framework on which one embodiment of the present invention may be implemented;

Figure **5** shows a representative hardware environment associated with the various devices, i.e. host, etc. shown in the network diagram of Figure **4**;

Figure **6** illustrates an exemplary data block;

Figure **7** illustrates a flow diagram of a method for further rolling up the services in a single data block; and

Figures **8-11B** illustrate an alternate exemplary architecture with which the foregoing techniques may be implemented.

Description of the Preferred Embodiments

Figures **1-2** illustrate the prior art. Figure **3** illustrates a method **300** for generating a single record reflecting multiple services for accounting purposes. First, in operation **302**, a plurality of services carried out over a network, i.e. the Internet, are identified. Such services may include a hypertext transfer protocol (HTTP) session, an electronic mail session, a multimedia streaming session, and/or a voice over Internet Protocol (IP) session.

Thereafter, data is collected describing each of the plurality of services in operation **304**. Examples of such data may include, but are not limited to a session's source, destination, user name, duration, time, date, type of server, QoS information, volume of data transferred, etc. A single record is subsequently generated including the collected data. Such single record represents each of the services. Note operation **306**. As an option, the single record may include data in various prior art data block formats discussed hereinabove in the background section. Thus, the single record may include a combination of different data block formats.

Figure **4** illustrates an exemplary network framework **400** on which one embodiment of the present invention may be implemented. It should be noted that the network framework **400** of Figure **4** need not necessarily be used, and any type of network framework may be utilized per the desires of the user. As shown in Figure **4**, various network components may be provided including a router **402** for routing information between various portions of the network. In one embodiment, such network may include the Internet using a communication protocol such as TCP/IP or IPX. It should be noted, however, that the network may include any type of network including, but not limited to a wide area network (WAN), Metropolitan Area Network (MAN), local area network (LAN), etc.

Further provided is a host **404** coupled to the router **402** for sending information thereto and receiving information therefrom. A firewall **406** may also be coupled to router **402** for controlling access to a network or a plurality of interconnected devices **408**. While various network components have been disclosed, it should be understood that the present invention may be implemented in the context of any type of network architecture and in any type of network device such as proxy servers, mail servers, hubs, directory servers, application servers, AAA (Authentication, Authorization, Accounting) servers, etc.

Coupled to the various network devices is an aggregator **410**. In use, the aggregator **410** receives records from the devices for the purpose of aggregating the same. In the present description, aggregation refers to consolidation, analysis, or any other type of handling of data. Once aggregated, the records may be used to afford any desired type of service, OSS (Operational Support System), and/or BSS (Business Support System), i.e. billing, fraud detection, network monitoring, traffic engineering, etc. By this structure, the services may be identified, and data collected in accordance with operations **302** and **304**.

Figure **5** shows a representative hardware environment associated with the various devices, i.e. host, etc. shown in the network diagram of Figure **4**. Such figure illustrates a typical hardware configuration of a workstation in accordance with a preferred embodiment having one or multiple central processing units **510**, such as a microprocessor, and a number of other units interconnected via a system bus **512**. The workstation shown in Figure **5** includes a Random Access Memory (RAM) **514**, Read Only Memory (ROM) **516**, an I/O adapter **518** for connecting peripheral devices such as disk storage units **520** to the bus **512**, a user interface adapter **522** for connecting a keyboard **524**, a mouse **526**, a speaker **528**, a microphone **532**, and/or other user interface devices such as a touch screen (not shown) to the bus **512**, communication adapter **534** for connecting the workstation to a communication network **535** (e.g., a data processing network) and a display adapter **536** for connecting the bus **512** to a display device **538**.

The workstation may have resident thereon an operating system such as the Microsoft Windows NT or Windows Operating System (OS), the IBM OS/2 operating system, the MAC OS, or UNIX operating system. It will be appreciated that a preferred embodiment may also be implemented on platforms and operating systems other than those mentioned. A preferred embodiment may be written using JAVA, C, and/or C++ language, or other programming languages, along with an object oriented programming methodology. Object oriented programming (OOP) has become increasingly used to develop complex applications.

For further information on another exemplary architecture embodiment, reference may be made to PCT application WO9927556A2 entitled "NETWORK ACCOUNTING AND BILLING SYSTEM AND METHOD" published June 3, 1999, which is incorporated herein by reference in its entirety. More information on such exemplary system will be set forth hereinafter starting with reference to Figure **8**.

It should be noted that the foregoing architectures should not be construed as limiting in any manner, and should be taken to merely represent exemplary systems for illustrative purposes only. For example, the present embodiment may be implemented in the context of any chip, host, router, network device, architecture, etc. that is desired.

Figure **6** illustrates an exemplary data block **600** generated in accordance with operation **306**. As shown, the data block **600** may include data collected during a plurality of distinct services, i.e. HTTP session, an electronic mail session, and/or a voice over IP session. The data may be directly collected from network devices or obtained from aggregators as illustrated in Figure **4**. For example, the data block

600 may include an account identifier, a start time of a session, a duration of the session, a number of HTTP bytes downloaded during the session, mail bytes, etc.

Figure **7** illustrates a flow diagram of a method **700** for further rolling up the services into a single data block. Such single data block may be sent to a Business Support System (BSS)for the purposes of billing at least one recipient of the services, or engaging in other accounting services.

As shown in Figure **7**, a plurality of the records **702** may be collected and grouped, where each group of records relates to the usage of a specific type of service, e.g. web surfing, e-mail, voice over IP calls, and multimedia streaming, etc. The records **702** may reflect the usage of any granularity required for billing of a BSS. Thereafter, tables **703** may be employed to identify customers who received the services identified in the records **702**. This may be accomplished by correlating an IP address with user identifiers, users' location information, company identifiers, or any other desired method.

Thereafter, separate records **704** may be generated based upon correlating a plurality of records **702** and information contained in tables **703**. How the correlation is performed may depend on the billing requirements of a BSS. Such separate records **704** may include a company identifier and usage data associated with one particular service. As such, the separate record **704** may represent each of the plurality of records **702**.

Still yet, a rolled up record **706** may include data of all of the services associated with each of the particular companies. Accordingly, this technique may be used to roll up a plurality of records associated with a plurality of customers, and services provided to those customers. The structure of such a record **706** is totally flexible and may include any type of customer/company identifier, and associated usage information of a service.

In one specific example of use, the IP usage information of individual mobile users (typically associated with an IP address) may be collected and stored in probe records **702**, web proxy records **704**, and voIP records **706**. By correlating and aggregating the IP usage information with user information stored in an LDAP, GPRS (General Packet Radio Service), and database, aggregated IP usage information may be obtained that are stored in aggregated netflow records, aggregated web proxy records, and aggregated voIP records. Based on the aggregated usage information, a table may be constructed in the manner shown to present a real-time view of the total resource consumption for all multi-party customers. In the context of the present description, a "real-time" environment is that which ensures no more than a fixed latency.

The present invention offers great flexibility to support billing requirements for various business models. It may even be the key enabler to support advanced billing schemes, such as a marketing tariff/rate scheme. A single consolidated record that rolls up information related to services, e-business transactions, content accesses, and information inquiries, etc. can be provided to a BSS for billing. By this design, schemes may be supported where, for example, a customer gets free Internet access while he or she is buying books from an affiliated e-store. The present invention thus allows usage data to be processed close to a collection point by organizing it in a unique manner, namely a single record.

<u>Alternate Exemplary Embodiment</u>

One embodiment of a system in which the foregoing details may be implemented will now be set forth. Of course, the following description should not be construed as limiting in any manner, and should be taken to represent merely an exemplary system for illustrative purposes.

The present embodiment includes a multi-source, multi-layer network usage metering and mediation solution that gives Network Service Providers (NSPs), including Internet Service Providers (ISPs) and enterprise network (Intranet) operators, the information needed to set the right-price for IP(Internet Protocol) services. With the system, the providers can generate accurate usage-based billing and implement usage-based charge-back models. The system derives IP session and transaction information, collected in real time, from a multitude of network elements. The system gathers, correlates, and transforms data from routers, switches, firewalls, authentication servers, LDAP, Web hosts, DNS, and other devices to create comprehensive usage and billing records.

The system transforms raw transaction data from network devices into useful billing records though policy-based filtering, aggregation, and merging. The result is a set of detail records (DRs). In some embodiments, the detail records are XaCCT Detail Records (XDRs™) available from XaCCT Technologies. DRs are somewhat similar in concept to the telephony industry's Call Detail Records (CDRs). Thus, DRs can be easily integrated with existing Customer Care and Billing (CCB) systems.

In addition to billing data, DRs enable NSPs to deploy new services based on documented usage trends, plan network resource provisioning, and audit service usage. The system provides a clear picture of user-level network service use by tracking a variety of metrics such as actual session Quality of Service (QoS), traffic routes, and end-user application transactions.

The system is based on a modular, distributed, highly scalable architecture capable of running on multiple platforms. Data collection and management is designed for efficiency to minimize impact on the network and system resources.

The system minimizes network impact by collecting and processing data close to its source. Modular architecture provides maximum configuration flexibility, and compatibility with multiple network information sources.

The system, or other embodiments, may have one or more of the following features.

Data collection can be from a wide range of network devices and services, spanning all layers of the network - from the physical layer to the application layer.

Real-time, policy-based filtering, aggregation, enhancement and merging create accurate, detailed and comprehensive session detail records(DRs).

Real time correlation of data from various sources allows billing record enhancement.

Leverages existing investment through integration with any customer care & billing solution, reducing costs, minimizing risks and shortened time-to-market.

Non-intrusive operation eliminates any disruption of network elements or services.

Web-based user interface allows off-the-shelf browsers to access the system, on-demand, locally or remotely.

Carrier-class scalability allows expansion to fit an NSPs needs without costly reconfiguration.

Distributed filtering and aggregation eliminates system capacity bottlenecks.

Efficient, centralized system administration allows on-the-fly system reconfigurations and field upgrades.

Customized reporting with built-in report generation or an NSPs choice of off-the-shelf graphical reporting packages.

Comprehensive network security features allow secure communication between system components and multiple levels of restricted access.

System Details

The following describes the system **800** of Figure **8**. The system **800** allows NSPs to account for and bill for IP network communications. The following paragraphs first list the elements of Figure **8**, then describes those elements and then describes how the elements work together. Importantly, the distributed data gathering, filtering and enhancements performed in the system **800** enables load distribution. Granular data can reside in the peripheries of the system **800**, close to the information sources. This helps avoids reduce congestion in network bottlenecks but still allows the data to be accessible from a central location. In previous systems, all the network information flows to one location, making it very difficult to keep up with the massive record flows from the network devices and requiring huge databases.

The following lists the elements of Figure **8**. Figure **8** includes a number of information source modules (ISMs) including an ISM **810**, an ISM **820**, an ISM **830**, an ISM **836**, an ISM **840**, and an ISM **850**. The system also includes a number of network devices, such as a proxy server **801**, a DNS **802**, a firewall **803**, an LDAP **806**, a CISCO NetFlow **804**, and a RADIUS **805**. The system also includes a number of gatherers, such as a gatherer **867**, a gatherer **862**, a gatherer **863**, a gatherer **864**, and a gatherer **865**. The system of Figure 8 also includes a central event manager (CEM) **870** and a central database (repository) **875**. The system also includes a user interface server **885** and a number terminals or clients **880**.

This paragraph describes how the elements of Figure **8** are coupled. The various network devices represent devices coupled to an IP network such as the Internet. The network devices perform various functions, such as the proxy server

801 providing proxy service for a number of clients. Each network device is coupled to a corresponding ISM. For example, the proxy server **801** is coupled to the ISM **810**. The DNS **802** is coupled to the ISM **820**. The firewall**803** is coupled to the ISM **830**. The ISM **836** is coupled to the LDAP **806**. The ISM **840** is coupled to the CISCO NetFlow **804**. The ISM **850** is coupled to the RADIUS **805**. Each gatherer is associated with at least one ISM. Thus, the gatherer **861** is associated with the ISM **810** and is therefore coupled to that ISM. The gatherer **862** is coupled to the ISM **820**. The gatherer **863** is coupled to the ISM **830** and the ISM **836**. The gatherer **864** is coupled to the ISM **840**. The gatherer **865** is coupled to the ISM **850**. The various gatherers are coupled to the CEM **870**. The user interface server is coupled to the terminals **880** and the CEM **870**.

The following paragraphs describe each of the various elements of Figure **8**.

Network Devices

The network devices represent any devices that could be included in a network. (Throughout the description, a network device, unless specifically noted otherwise, also refers to an application server.) A network device represents a subset of information sources that can be used by the system **800**. That is, the network devices are merely representative of the types of sources of information that could be accessed. Other devices such as on-line transaction processing databases can be accessed in other embodiments of the invention. Typically, the network devices keep logging and statistical information about their activity. A network information source can be the log file of a mail server, the logging facility of a firewall, a traffics statistics table available on a router and accessible through SNMP, a database entry accessible through the Internet, an authentication server's query interface, etc. The network devices represent the   information sources accessed by the ISMs.

Each type of network device can be accessing using a different method or protocols. Some generate logs while others are accessible via SNMP, others have proprietary APIs or use other protocols.

ISMs

The ISMs act as an interface between the gatherers and the network devices enabling the gatherers to collect data from the network devices. Thus, the ISMs represent modular, abstract interfaces that are designed to be platform-neutral. The information source modules act as interfaces or "translators", sending IP usage data, in real time, from the network devices to the gatherers. Each ISM is designed for a specific type of network data source. (In other embodiments, some ISMs are generic in that they can extract information from multiple network devices). ISMs can be packaged separately, allowing NSPs to customize ISM configurations to meet the specific requirements of their network. For example, in the system of Figure **8**, if the NSP did not have Cisco NetFlow devices, then the ISM **840** would not have to be included.

The ISMs can communicate with its corresponding network device using protocols and formats such as UDP/IP, TCP/IP, SNMP, telnet, file access, ODBC, native API, and others.

In some embodiments, the reliability of system **800** is enhanced through on-the-fly dynamic reconfiguration, allowing the NSP to add or remove modules without disrupting ongoing operations. In these embodiments, the CEM **870** can automatically update the ISMs.

The following ISMs are available in some embodiments of the invention.

- Categorizer - Classifies a session to a category according to user-defined Boolean expression.

- DNS (e.g. ISM **820**) - Resolves host names and IP addresses.
- Generic Proxy Server (e.g., ISM **810**) - Collects data from access logs in a common log format.
- Port / Protocol Resolution - Converts protocol/port information to account names and vice versa.
- CheckPoint FireWall- 1 -Collects data from FireWall- 1 accounting log and security log.
- Cisco IOS IP Accounting - Collects accounting data from a Cisco router using IOS IP accounting.
- Cisco NetFlow Switching - Collects session data from a Cisco router via NetFlow switching.
- NETRANET - Collects information from a standard network device.
- Netscape Proxy Server - Collects data from a Netscape Proxy Server.
- Microsoft Proxy Server - Collects data from a Microsoft ProxyServer.

ISMs can be synchronous, asynchronous or pipe. The data from an asynchronous ISM is dynamic so that the asynchronous ISM reacts to the information and relays it to the associated gatherer without prompting from other information sources in the system **800**. If the firewall **803** were a CheckPoint FireWall-1, then the ISM **830** would be an example of an asynchronous ISM. When a network session is initiated, the details are recorded by the FireWall-1 **803**. The corresponding ISM **830** receives the details and passes them on automatically to the gatherer **863**.

Synchronous ISMs provide its information only when accessed by a gatherer. The ISM **820** is an example of a synchronous ISM. The DNS server**802** maintains information matching the IP addresses of host computers to their domain addresses. The ISM **820** accesses the DNS server **802** only when the ISM **820** receives a request from the gather **862**. When the DNS server **802** returns a reply, the ISM **820** relays the reply information to the gatherer **862**.

Pipe ISMs operate on record flows (batches of records received from information sources). Pipe ISMs process one or more enhancement flows the records as the flows arrive. The pipe ISM may initiate new record flows or may do other things such as generate alerts or provision network elements to provide or stop services. The pipe is implemented as an ISM to keep the internal coherency and logic of the architecture. (Record flows can terminate in a database or in a pipe ISM. The pipe ISM can perform filtering and aggregation, send alarms, or act as a mediation system to provision network elements when some event occurs or some accumulated value is surpassed. Specifically, pipe ISMs can act to enable pre-payment systems to disable certain services such as a voice IP call, when the time limit is surpassed or amount of data is reached.)

The gatherers can include caches and buffers for storing information from the ISMs. The buffers allow the gatherers to compensate for situations where there is a loss of connection with the rest of the system **800**. The cache sizes can be remotely configured. The cache minimizes the number of accesses to the Information Source.

ISM queries can be cached and parallelized. Caching of synchronous ISM queries provides for fast responses. Parallelizing queries allows for multiple queries to be processed at the same time.

<u>Gatherers</u>

The gatherers gather the information from the ISMs. In some embodiments, the gatherers are multi-threaded, lightweight, smart agents that run on non-dedicated hosts, as a normal user application on Windows NT or Unix, as a background process, or daemon. What is important though is that the gatherers can be any hardware and/or software that perform the functions of a gatherer.

The gatherers can be installed on the same network segment as the network device such as router and switch or on the application server itself. This placement of a gatherer minimizes the data traffic impact on the network.

The gatherers collect network session data from one or more ISMs. Session data can be sent to another gatherer for enhancement or to the CEM **870** for merging and storing in the central database **870**. The gatherers can be deployed on an as needed basis for optimal scalability and flexibility.

The gatherers perform flexible, policy-based data aggregation. Importantly, the various types of ISMs provide different data and in different formats. The gatherers normalize the data by extracting the fields needed by the CEM **870** and filling in any fields that may be missing. Thus, the gatherers act as a distributed filtering and aggregation system. The distributed data filtering and aggregation eliminates capacity bottlenecks improving the scalability and efficiency of the system **800** by reducing the volume of data sent on the network to the CEM **870**.

Aggregation can be done by accumulating groups of data record flows, generating a single data record for each group. That single record then includes the aggregated information. This reduces the flow of the data records.

Filtering means discarding any record that belongs to a group of unneeded data records. Data records are unneeded if they are known to be collected elsewhere. A policy framework enables the NSP to configure what to collect where.

Filtering and/or aggregation can be done at any point along a data enhancement (described below) so that aggregation schemes can be based on enhanced data records as they are accumulated. The filtering and/or aggregation points are treated by the system **800** as pipe ISMs which are flow termination and flow starting points (i.e.: like an asynchronous ISM on the starting end and like a database on the terminating end). Data enhancement paths and filtering and/or aggregation schemes can be based on accumulated parameters such as user identification information and a user's contract type.

As noted above, the PISM can be used in the context of filtering and/or aggregation. One or more record flows can terminate at the PISM and can be converted into one or more new record flows. Record flows are grouped based on matching rules that apply to some of the fields in the record flows, while others are accumulated or undergo some other operation such as “maximum” “average”. Once the groups of accumulated records have reached some threshold, new accumulated records are output. This can be used for example in order to achieve a business-hybrid filtering and aggregation data reduction by imposing the business rules or the usage-based products that are offered to the customer, onto the record flows as they are collected in real-time. This is done instead of previous system where the information is stored in a database and then database operations are performed in order to create bills or reports. The filtering and aggregation reduces the amount of data that is stored in the central database **875** while not jeopardizing the granularity of data that is necessary in order to create creative usage-based products.

Typically, data collected from a single source does not contain all the information needed for billing and accounting, such as user name and organization. In such cases, the data is enhanced. By combining IP session data from multiple sources, such as authentication servers, DHCP and Domain Name servers, the gatherers create meaningful session records tailored to the NSP's specific requirements. In the example of Figure **8**, the gatherer **861** can provide information to the gatherer **862** so that the source IP address for an Internet session from the proxy server **801** can be combined with the domain address from the DNS server **802**.

The enhancement procedure can be triggered by an asynchronous ISM. The information from the asynchronous ISM is associated with field enhancements in the central database **875**. A field enhancement defines how a field in the central database is filled from the source data obtained from the asynchronous ISM. Through the field enhancements, the missing parameters are added to a record using the data collected from one or more synchronous ISMs. Enhancements are described in detail below.

The gatherers can include caches and buffers for storing information from the ISMs. The buffers allow the gatherers to compensate for situations where there is a loss of connection with the rest of the system **800**. The caches can reduce the number of accesses to an information source. The buffer and/or cache sizes can be remotely configured.

Central Event Manager (CEM)

The Central Event Manager (CEM) **870** acts as the central nervous system of the system **800**, providing centralized, efficient management and controls of the gatherers and the ISMs. The CEM **870** can perform one or more of the following tasks.

- Coordinates, controls, and manages the data collection process. The CEM **870** coordinates the operation of the gatherers and manages the flow of data through the system **800** through the collection scheme defined in the system configuration. The latter includes the configuration of the gatherers, the ISMs, the network devices, the fields in the central database **875** (described below), and the enhancement procedures. Based on the collection scheme the CEM **870** determines the system **800**'s *computation flow* (the set of operations the system **800** must perform to obtain the desired information). The CEM **870** then controls all the gatherers, instructing them to perform, in a particular sequence, the operations defined in the computation flow. The CEM **870** receives the records collected by the gatherers and stores them in the central database875. NSPs can configure the CEM **870** to *merge* duplicate records before storing them in the central database **875**. Record merging is described below.

- Performs clean-up and aging procedures in the database **875**. The system **800** collects and stores large amounts of session information every day. The CEM **870** removes old data to free space for new data periodically. The NSP defines the expiration period for the removal of old records. The CEM **870** is responsible for coordinating the removal of records from the central database **875**. The CEM **870** places a time stamp on every record when the record enters the central database **875** and deletes the record after the time period the NSP has defined elapses.
- Provides centralized system-wide upgrade, licensing, and data security. The NSP can perform version upgrades of the system **800** at the CEM **870**. The gatherers can be automatically upgraded once a new version is installed on the host computer of the CEM **870**. ISMs are also installed via the CEM **870** and exported to the gatherers. The CEM **870** maintains a list of licenses installed in the system and verifies periodically if the system is properly licensed. This feature lets the NSP centrally install and uninstall licenses. It also prevents unlicensed use of the system **800** and any of its components.
- Monitors the state of the gatherers and ISMs. The gatherers periodically communicate with the CEM **870**. The CEM **870** continuously monitors the state of each gatherer and network devices in the system **800**. The CEM **870** can be fault-tolerant, that is, it can recover from any system crash. It coordinates the recovery of the system **800** to its previous state.

In some embodiments, a key directory server is associated with the CEM **870**. To transfer less data between the elements of the system **800**, it is desirable that each piece of data to carry little descriptive data. For example, if IP address data is transferred between a gatherer and the CEM **870**, a description of the IP address data is typically included. In some embodiments, data name/key, type, and length descriptions are included with the actual IP address data. In other embodiments, there the key directory server reduces the amount of descriptive information being sent. Every key in the directory server has a type and a length. Fields can be identified as variable length. Therefore, data type information need not be transmitted between elements in the system **800** if the elements use a common reference key stored in the directory server. Returning to the IP address data, by using the key directory server, elements need only send two bytes for the key id and four bytes for the actual address. Most of the data being sent in the system is relatively short in length. Therefore, the directory server helps reduce the amount of information being sent between the elements in the system **800**.

Keys can be added to the directory server. The directory server can therefore support expansion of the kinds of fields being sent by allowing system elements to update their locally stored key ids. For example, after a recipient receives a record with an "unknown" key, it contacts the directory server to get the key definition.

Central Database

The central database **875** is the optional central repository of the information collected by the system **800**. The central database **875** is but one example of a sink for the data generated in the system **800**. Other embodiments include other configurations. The central database **875** stores and maintains the data collected by the gatherers, as well as the information on the configuration of the system **800**. Thus, in configuring the system 800, the NSP defines what data will be stored in each field in the central database **875** and how that data is collected from the ISMs.

The information on network sessions is stored in the database in the form of a table. Each field in the table represents a network session parameter. Each record describes a network session. The system **800** has a set of pre-defined fields that are configured by the CEM **870** on installation. The NSP can modify the central database **875** structure by adding, deleting, or modifying fields. The NSP access the data in the central database **875** by running queries and reports. The old data is removed from the central database **875** to free space for new data periodically. You can specify the time interval for which records are stored in the central database **875**. The structure of the central database **875** with some of the predefined fields is illustrated in the following figure.

As each IP session may generate multiple transaction records, during the merge process the CEM **870** identifies and discards duplications, enhancing the efficiency of the data repository. Generally, data records are passed through the merger program, in the CEM **870**, into the central database **875**. However, the data records are also cached so that if matching records appear at some point, the already stored records can be replaced or enhanced with the new records. The database tables that contain the record flows can be indexed, enhancing the efficiency of the data repository. A merge is achieved by matching some of the fields in a data record and then merging the matching records from at least two record flows, transforming them into one record before updating the central database **875**. In some embodiments, adaptive tolerance is used to match records. Adaptive tolerance allows for a variation in the values of fields that are compared (e.g., the time field value may be allowed to differ by some amount, but still be considered a match). The adaptive aspect of the matching can include learning the appropriate period to allow for the tolerance. The reason that the records that do not match any previous records are sent through into the central database **875**, in addition to being cached for later matching, is to avoid loss of data in case of system failure.

The system **800** supports a non-proprietary database format enabling the central database **875** to run on any of a number of commercially available databases (e.g., MS-SQL Server, Oracle Server, D132, etc.).

<u>User Interface Server and Clients</u>

The User Interface Server (UIS) **885** allows multiple clients (e.g. terminals **880**) to access the system **800** through, the Microsoft Internet Explorer with Java™

Plug-in or Netscape Navigator with Java™ Plug-in. Other embodiments can use other applications to access the system **800**. The main function of the UIS **885** is to provide remote and local platform independent control for the system **800**. The UIS **885** can provide these functions through windows that correspond to the various components of the system **800**. Access to the system **800** can be password protected, allowing only authorized users to log in to the system and protecting sensitive information.

The NSP can perform one or more of the following main tasks through the UIS **885**:

- Configure the system **800**.
- Create and run queries and reports on network activity and resource consumption.
- Register and license the system **800**.

Data Distillation

Figure **9** illustrates the data distillation process performed by the system of Figure **7**. The data distillation aggregates and correlates information from many different network devices to compile data useful in billing and network accounting.

First, the ISMs **910** gather data from their corresponding network device. Note that for some ISMs (e.g. pipe ISMs), real-time, policy-based filtering and aggregation **915** can also be done. This data is then fed to the gatherers **920**. The gatherers **920** perform data enhancement to complete the data from the ISMs **910**. The results are provided to the CEM **870**. The CEM **870** performs data merges **970** to remove redundant data. The merged data is then optionally stored in the central database **875** as a billing record **975** or is sent directly to an external system. The billing record information can be accessed from external applications, through the application interface **990**, via a data record **980**. Filtering and/aggregation and/or data enhancements can be done at any stage in the system **800**.

<u>Data Enhancement</u>

As mentioned above, the gatherers **920** provide data enhancement features to complete information received from the ISMs **910**. The following describes some example data enhancement techniques used in some embodiments of the invention.

Figure **10** illustrates an example of data enhancement. Data enhancement comprises a number of field enhancements. A field enhancement specifies how the data obtained from the trigger of the enhancement procedure is processed before it is placed in a single field in the central database **875**. The data can be placed in the field directly, or new information may be added to the record by applying a Synchronous ISM function. (In the example below, the function resolves the IP address to a host FQDN"). Field enhancements may involve one or multiple steps. There is no limit to the number of steps in a Field Enhancement. The data record starts with fields obtained from an asynchronous ISM **1000**. The fields in the DR **1000** are then enhanced using the field enhancements. The enhanced fields result in the DR **1020**.

A visual representation of an enhancement can be presented to the NSP. The enhancement may include an itinerary of ISMs starting off with an AISM, passing through PISMs, and terminating in the CEM **870**. Using this view of the system **800**, the NSP need not be shown the actual flow of data since the flow may be optimized later in order to achieve better performance. This is more of a graphical logical view of how the enhancement is achieved in steps. (PISMs can terminate more than one flow and initiate more than one flow.)

A visual representation of a field enhancement shows the per-field flow of data correlation. This process ends in the CEM **870** or in a PISM. The NSP supplies information telling the system **800** how to reach each of the terminating fields (in the CEM **870** or the PISM) starting off from the initiating fields (PISM or AISM). Each step of enhancement defines cross correlation with some SISM function.

Figure **11A** illustrates various field enhancements (**1110** through **1140**). A field enhancement includes applying zero or more functions to a field before storing the field in a specified field in the central database **875**.

One-step Field Enhancement **1110**. The initial source data from the asynchronous ISM is placed directly in a field in the central database **875**. Example: the field enhancement for the Source IP field.

Two-step Field Enhancement **1120**. The initial source data from the asynchronous ISM is used to obtain new additional data from a synchronous network device and the new data is placed in a field in the central database **875**. Example: the field enhancement for the Source Host field.

Three-step Enhancement **1130**. The initial source data from the asynchronous ISM is used to obtain additional data from a synchronous ISM. The result is used to obtain more data from another ISM and the result is placed in a field in the central database **875**.

The following illustrates an example data enhancement. Suppose the data obtained from a proxy server **801** contains the source IP address of a given session, such as 199.203.132.2, but not the complete domain address of the host computer (its Fully Qualified Domain Name), such as www.xacct.com. The name of the host can be obtained by another network device - the Domain Name System (DNS **802**) server. The DNS server **802** contains information that matches IP addresses of host computers to their Fully Qualified Domain Names (FQDNs). Through an enhancement procedure the information collected from the proxy server **801** can be supplemented by the information from the DNS**802**. Therefore, the name of the host is added to the data (the data record) collected from the proxy server **801**. The process of adding new data to the data record from different network devices can be repeated several times until all required data is collected and the data record is placed in the central database **875**.

Figure **11B** illustrates another example data enhancement where an enhanced record **1190** is created from an initial netflow record **1192**. Fields in the enhanced record **1190** are enhanced from the radius record **1194**, the QoS policy server record **1196**, the NMS DI3 record **1198**, and the LDAP record **1199**.

Defining Enhancement Procedures

The following describes the process for defining enhancement procedures in some embodiments of the system. Typically defining an enhancement procedure for the system **800** includes (1) defining enhancement procedures for each asynchronous ISM and (2) configuring field enhancements for all fields in the central database **875** for which the NSP wants to collect data originating from an asynchronous ISM that triggers the corresponding enhancement procedure.

An enhancement procedure can be defined as follows.

1. Access the CEM 870 using the UIS **880**.
2. Select the enhancement procedures list using the UIS **880**.
3. Define the name of the new enhancement procedure.
4. Select a trigger for the new enhancement procedure. The trigger can correspond to any asynchronous ISM in the system **800**. Alternatively, the trigger can correspond to any asynchronous ISM in the system **800** that has not already been assigned to an enhancement procedure.
5. Optionally, a description for the enhancement procedure can be provided.

6. The new enhancement procedure can then be automatically populated with the existing fields in the central database **875**. Optionally, the NSP can define the fields (which could then be propagated to the central database **875**). Alternatively, based upon the type of asynchronous ISM, a preset set of fields could be proposed to the NSP for editing. What is important is that the NSP can define field procedures to enhance the data being put into the data records of the central database **875**.

7. The NSP can then define the field enhancements for every field in the new enhancement procedure for which the NSP wants to collect data from the ISM that is the trigger of the new enhancement procedure.

Defining Field Enhancements

Defining a field enhancement involves specifying the set of rules used to fill a database field from the information obtained from the trigger of the enhancement procedure. The NSP defines field enhancements for each field in which NSP wants to collect data from the trigger. If no field enhancements are defined, no data from the trigger will be collected in the fields. For example, suppose the firewall asynchronous ISM **830** that triggers an enhancement procedure. Suppose the central database **875** has the following fields: source IP, source host, destination IP, destination host, user name, total bytes, service, date/time, and URL If the NSP wants to collect session data for each field except the URL from the firewall ISM **830**, which triggers the enhancement procedure, the NSP defines a field enhancement for each field with the exception of the URL.

In some embodiments, the field enhancements are part of the enhancement procedure and the NSP can only define and modify them when the enhancement procedure is not enabled.

The field enhancements can be defined in a field enhancement configuration dialog box. The field enhancement configuration dialog box can have two panes. The first displays the name of the enhancement procedure, the name of its trigger, and the name and data type of the field for which the NSP is defining the field enhancement. The second is dynamic and interactive. Its content changes depending on the NSP's input. When first displayed, it has two toggle buttons, End and Continue, and a list next to them. The content of the list depends on the button depressed.

When End is depressed, the list contains all output fields whose data type matches the data type of the field for which the NSP is defining the field enhancement. For example, if the field's data type is IP Address, the list contains all fields that are of the same type, such as source IP and destination IP that the AISM supplies. The fields in the list can come from two sources: (1) the source data which the gatherer receives from the trigger and (2) the result obtained by applying a synchronous ISM function as a preceding step in the field enhancement. The following notation is used for the fields:

OutputFieldName for the output of a field origination from the trigger

SISName. FunctionName (InputArgument). OutputField for the output of a field that is the result of applying a function

SISName ... OutputField for the output of a field that is the result of applying a function as the final step of a field enhancement. The following examples are presented.

Source IP is the field provided by the trigger of the enhancement procedure that contains the IP address of the source host.

DNS ... Host Name and DNS.Name(Source IP).Host name are the names of a field originating from the resolved function Name of a network device called DNS that resolves the IP address to a domain address. The input argument of the function is the field provided by the trigger of the enhancement procedure, called source IP. It contains the IP address of the source host. The function returns the output field called Host Name that contains the domain address of the source host. The notation DNS ... Host Name is used when the field is the result of applying the function as the final step of a field enhancement. The notation is DNS.Name(Source IP).Host Name is used when the field is used as the input to another function.

In the user interface, if End is unavailable, none of the output fields matches the data type of the field.

When Continue is depressed, the list contains all applicable functions of the available synchronous network device configured in the system **800**. If the preceding output does not match the input to a function, it cannot be applied and does not appear on the list.

The following notation is used for the functions.

SISName.FunctionName(InputFieldName:InputFieldDataType)(OutputFieldName.-OutputFieldDataType)

When the function has multiple input and/or output arguments, the notation reflects this. The arguments are separated by commas.

The following example shows a field enhancement.

DNS. Address(Host Name:String) -> (IP Address:IP Address)

Where DNS is the name of the synchronous ISM (or network device) as it appears in the system configuration.

Address is the name of the function.

(Host Name:String) is the input to the function - host FQDN of data typeString

(IP Address:IP Address) is the output - IP address of data type IPAddress

The NSP can define the field enhancement by choosing items from the list. The list contains the option <none> when the End button is depressed. Choosing this option has the same effect as not defining a field enhancement: no data from the trigger will be stored in the field in the central database **875**.

Additional Embodiments

The following describes additional embodiments of the invention.

In some embodiments, the user interface used by an NSP to configure the system **800** can be presented as a graphical representation of the data enhancement process. Every step in the enhancement can be shown as a block joined to another block (or icon or some graphical representation). The properties of a block define the operations within the block. In some embodiments, the entire data enhancement process from network devices to the central database **875** can be shown by linked graphics where the properties of a graphic are the properties of the enhancement at that stage.

In some embodiments, multiple CEMs **870** and/or central databases **875** can be used as data sources (back ends) for datamart or other databases or applications (e.g., customer care and billing systems).

In some embodiments, the types of databases used are not necessarily relational. Object databases or other databases can be used.

In some embodiments, other platforms are used. Although the above description of the system **800** has been IP network focused with Unix or Windows NT systems supporting the elements, other networks (non-IP networks) and computer platforms can be used. What is important is that some sort of processing and storing capability is available at the gatherers, the CEMs, the databases, and the user interface servers.

In some embodiments, the gatherers and other elements of the system **800**, can be remotely configured, while in other embodiments, some of the elements need to be configured directly. For example, a gatherer may not be remotely configurable, in which case, the NSP must interface directly with the computer running the gatherer.

In other embodiments, the general ideas described herein can be applied to other distributed data enhancement problems. For example, some embodiments of the invention could be used to perform data source extraction and data preparation for data warehousing applications. The gatherers would interface with ISMs that are designed to extract data from databases (or other data sources). The gatherers would perform filtering and aggregation depending upon the needs of the data mart (in such an embodiment, the central database and CEM could be replaced with/used with a data mart). The data enhancement.

While various embodiments have been described above, it should be understood that they have been presented by way of example only, and not limitation. Thus, the breadth and scope of a preferred embodiment should not be limited by any of the above-described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.